06 | 流量构建：流量平台如何选型？

2021-10-30 高楼

《全链路压测实战30讲》 课程介绍 >

讲述：高楼

时长 16:19 大小 14.95M

你好，我是高楼。

在第一篇文章中，我提到了全链路线上压测和传统线下压测的主要区别：

领资料

对比点	全链路线上压测	传统线下压测
流量	大（大概能到几十万、上百万、上千万/每秒的请求）	小（大概是几百、几千、几万/每秒的请求）
被测系统	大容量的线上系统	小容量的线下系统
硬件环境	线上环境	线下环境
软件环境	线上系统的软件配置	线下系统的软件配置
基础数据	脱敏的生产数据	造出来的数据
存储	生产环境存储	测试环境中的存储
网络结构	生产网络结构	测试网络结构

极客时间

既然全链路线上压测和传统线下压测有这么多差异，那它肯定会对我们的流量平台提出更高的要求。我们希望，一款全链路流量平台能够对整体线上环境进行全方位、真实、安全的压测，这样，我们才能够更好、更快、更精准地进行线上容量评估。

流量平台的需求

下面让我们来看看，全链路压测对流量平台的要求有哪些：

能够提供模拟线上真实流量的能力。也就是说，能够模拟真实流量，最好能够直接获取线上的真实流量进行压测。

能够实现海量数据的并发请求。这就要求平台的网络结构符合线上环境拓扑，特别是能满足有地域分布特点的 CDN 边缘节点需求。

压测类型需要支撑常用协议，如 RPC、HTTP 协议。这里也要提一下，我们这个课程的微服务电商项目都是标准的 RESTful 风格 HTTP 接口。

压测不能对线上用户的使用产生任何影响。压测流量的读请求不能对线上产生任何脏数据，也就是说，需要做到压测标记。

能够提供压测过程中的实时监控以及业务异常保护能力。线上压测是一项对风险管控要求更高的活动，压测平台需要有更及时的异常预警机制，也就是说必须要有压测实时监控和异常熔断能力。

值得注意的是，为了精准地获取到线上环境的容量，全链路压测有两个非常关键的点：

保证流量的真实性

保证环境的真实性

既然，全链路压测对我们的流量平台提出了这么多苛刻的要求，那么我们又如何基于项目实际需求进行选型呢？

保证流量的真实性

在流量构造方式上，根据主要的原理的不同，会采用不同的方式，下表整理了几种常见的方式：

常见方式	线上流量回放	线上流量引流	流量模拟
主要原理	将线上集群环境的流量复制多份发送到指定的仓库，然后使用指定压力机器进行放大回放，以达到压力测试的目的。	将线上集群环境中多台机器的请求转发到一台机器上，以达到压力测试的目的。	使用常见压测工具发起模拟流量调用来达到压力测试的目的。
常用手段	使用 TCPCopy、GoReplay、JVM-Sandbox-Repeater 等流量复制工具。	• 使用 Nginx 插件 ngx_http_mirror_module、Traefik Ingress 将线上集群中的流量复制转发到到一台机器上。 • 修改负载均衡中某台机器的权重，将更多的流量直接打到某台机器上，让待压测的机器分配更多的流量。	使用常见 JMeter、Locust、nGrinder、Gatling 等传统压测工具模拟。
优点	直接利用线上流量模型，使得压测的流量跟真实的业务流量更接近。	• 直接使用最真实的线上流量 • 操作方便 • 压测结果较为准确 • 不会产生脏数据	• 上手简单，学习成本较低 • 开源 • 改造成本低 • 性能开销较小
缺点	• 需要借助一些自动化工具来做提效 • 需要解决各种 Session 问题 • 需要处理写请求脏数据的问题 • 压力机器需要单独提供	只能满足单机的需求，无法满足整体集群的压测需求。	• 模拟线上真实流量模型成本较高 • 大部分情况都在内网使用 • 海量流量容易出现瓶颈 • 需要处理写请求脏数据问题
适应场景	• 业务链路比较长的系统 • 访问量大的系统需要改造	• 单机压测 • 基准场景	• 新系统上线 • 访问量不大的系统

极客时间

从上表我们可以看到，我们可以使用诸如 JMeter、Locust、nGrinder、Gatling 等传统压测工具模拟流量，也可以使用类似 TCPCopy、GoReplay、JVM-Sandbox-Repeater 等

流量复制工具，实时录制生产环境流量，并回放导向目标压力机器。同时，这些流量回放工具可以支持对真实流量进行放大或缩小。

于是有人说，就是因为这样，才应该直接用真实流量的方式去做嘛，这样就不用管流量模型了，直接就有生产的流量模型了。没错，只要你能通过生产流量放大回放的方式实现压力部分，确实可以不用考虑业务场景了。**但这么做的前提必须是，你录制的生产流量是可以覆盖全链路压测的业务场景。**

同时这里我也要批驳一个观点，**就是有些人觉得只有通过生产流量录制回放的方式，才是真实地模拟线上的流量。**事实上，这个观点是偏颇的。只能说，生产流量录制的方式，相比用传统的压测工具去模拟生产业务模型，看似在成本和复杂度上都更低。而实际上成本和复杂度上都不会减少。

我为什么这样说呢？

我的第二个专栏的第 6 讲中，主要描述了如何利用工具统计出业务模型，接下来，我会讲到怎样把这个业务模型配置到压测工具中去。

我刚开始以为，这应该是所有性能测试工程师的日常工作，没必要再啰嗦一遍了。但是，之后有越来越多的人问我这个知识点，我才发现，绝大部分性能测试工程师，并不清楚如何统计出业务模型，也不知道如何将它具体配置到压测工具中。

这就导致出现了一个严重的现实问题，**目前绝大部分企业在做的线下压测容量场景，都是瞎做一通。**即使线下压测的容量场景结果再怎么好看，也不能回答线上容量的问题。于是乎，大家都把目光转移到了全链路压测身上。

同时，影响流量真实性的一个重要的方面便是地域性分布，我们可以看到目前国内大部分的压测平台几乎都是只在内网使用的，除了那么几个互联网大厂，很少有涉及到考虑 CDN 分布的。从 DNS 负载均衡以及访问加速的架构角度来看，这块是不能忽略的。

最后，我再强调一下，由于流量的链路分布差异，流量数据的多样性，流量的地域性，只有努力让我们的项目压测流量更为“真实”，才能让容量场景压测结果更为精准。

保证环境的真实性

想想我们做性能压测的目标，通常都是这样的说法：保证线上系统的正常运行。而一个系统的整体容量，不仅有上层应用软件，还有硬件、网络、存储、负载均衡、防火墙等等一系列软硬件。当有些大厂线下不能完全满足这样的复杂的环境结构时，就只能后移到线上环境进行压测了。

于是就有人问了，是不是容量场景就一定得线上做呢？我的回答是，也不一定如此。

对于绝大部分的中小企业系统，部署架构并没有那么复杂。有条件的可以搭建一个 1:1 的镜像环境，没有条件的也可以通过一些手段做到大概的线上线下容量换算，当然这有着一定的门槛，这里就不展开了。有兴趣的话，可以参考我上个专栏讲过的《如何确定生产系统配置》，我们需要根据自己系统的特点，选择适合自己企业的方案，而不是一股脑地把所有的压测全搬到线上去进行。

流量工具的选型

说完这两个重要问题，我来说说如何选择流量工具。因为这次的目标是全链路压测，它包括基准场景、容量场景、稳定性场景等等，所以线上引流这种方式首先可以排除。因为线上引流只能满足单服务的基准场景或者是一个机房（集群）内的功能验证测试。

至于流量回放和流量模拟，二者都能满足我们的需求。流量回放工具的优势是不用管流量模型了，这就拉平了容量场景脚本编写的复杂度，但目前主流的流量回放工具都无法轻易解决 session 的问题，所以从系统安全的角度来说，工具或系统需要做对应的改造。

而我们这个专栏的落地项目，对系统的改造是可以接受的，因为流量模拟的方式我已经在《性能工程实战课》中详细介绍过了，在这里我就选择了线上流量回放的方式。

这里我给出几点建议：

> 相较于流量模拟，传统的压测工具学习成本相对较低，几乎所有的性能测试工程师都会掌握那么一两个，同时目前国内压测工具平台化改造的技术还是比较成熟的。关于 session 的问题，传统的压测工具的确也可以很好地解决。
>
> 如果系统链路不复杂，有平台化的需求，改造成本较高，那么还是推荐你使用传统的压测工具。

如果系统链路稍微有些复杂，能够承受改造成本，同时，它还有除了全链路压测之外其它的用途，比如线下回归测试之类的工作，那么我推荐你选择流量回放工具。

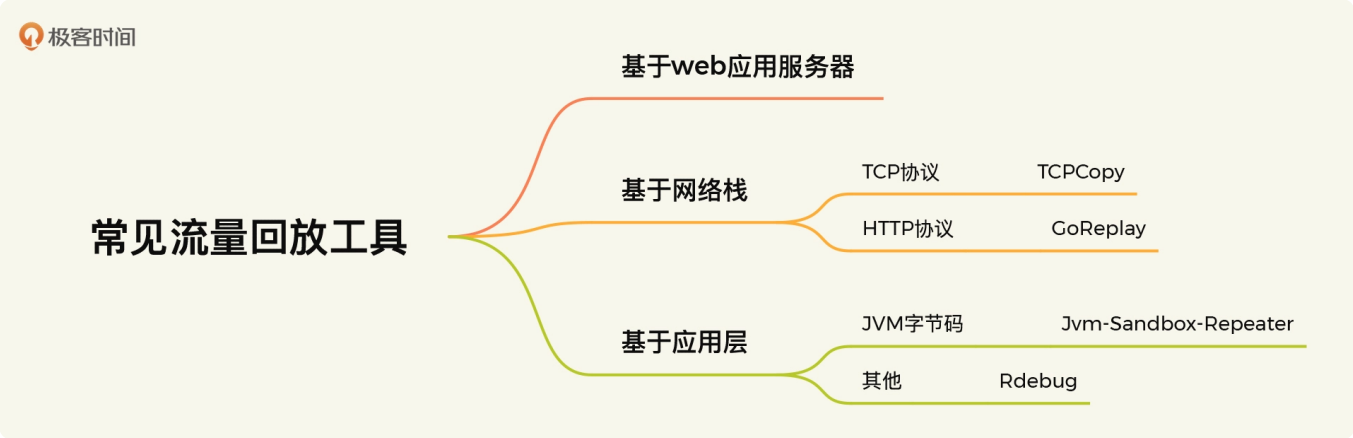
如果你只是想做线上基准场景，又或者是有线上灰度发布之类的需求，那么线上引流的这种方式是比较适合的。

如果你在一些特殊行业，比如说金融行业，跟钱有关的数据是受到严格监管的，从管理的角度上来说，线上流量回放和引流这两种方式几乎是不太可能的。

总之，不同方式的优劣都是相对的，我们只需要根据自己系统的需求及特点，选择适合自己的方式即可。

接下来，我们来看看常见的流量回放工具。

首先我们通过一个图梳理下常见的流量回放工具。



流量回放工具一般分成这几类：

1. 基于应用层的流量回放工具

   优点：实现简单，扩展性好，可定制化高。

   缺点：会挤占线上应用的资源（比如连接资源、内存资源等），还可能会因为耦合度高。而影响正常业务，需要一定的开发工作量，对工具本身的性能要求高。

2. 基于网络栈的流量回放工具，直接从链路层抓取数据包。

   优点：安装部署较为方便，对应用影响较小。

缺点：实现相对复杂一些，只适合无状态的业务，无法轻易处理 session。

3. 基于 web 服务器的请求复制。

优点：请求多样化、成本低。

缺点：不具备通用性、丢失网络延迟、占用在线资源比较严重。

熟悉了以上三种分类，我们再来梳理一下目前业界主流的几种开源流量回放工具：

常见工具	TCPCopy	GoReplay	Jvm-Sandbox-Repeater	RDebug
实现方式	底层依赖 pcap 库，复制基于 TCP 的 packets 包，修改 TCP/IP 头部信息，发送给测试服务器	底层依赖 pcap 库，启动一个 Gor 进程，它负责所有的工作包括监听、过滤和转发	基于JVM-Sandbox 框架，通过字节码增强注入对目标应用进行 AOP 拦截	启动一个录制进程，它负责所有的工作包括监听、过滤和转发
作者	王斌（网易）	Leonid Bugak（个人）	阿里巴巴	滴滴
开发语言	C	Golang	Java	Golang、C、C++、PHP
GitHub Stars	3.9k	14.1k	1.5k	1.1k
支持的协议	基于 TCP 的任意应用层协议	http(s)	HTTP/Java/Dubbo	HTTP/Thrift 等
支持的应用	所有应用	所有应用	仅 Java 应用	仅 PHP 应用
优点	• 支持的协议比较丰富 • 业务透明，代码完全解耦 • 支持保留原始请求	• 轻量，安装部署简单 • 上手简单且易用 • 功能丰富，支持流量过滤、倍数回放、参数改写等功能 • 业务透明，与业务代码完全解耦 • 提供插件机制，可扩展 • 知名度高，社区较为活跃	• 字节码增强方式，直接对 Java 类操作，功能较为强大 • 业务透明，与业务代码完全解耦 • 可插拔，较灵活，可自由组合	• 支持下游流量录制，回放不依赖下游环境。 • 业务透明，与业务代码完全解耦
缺点	• 部署结构稍复杂 • 部分丢包 • 有一定的学习成本 • 存在性能瓶颈 • 存在安全风险 • 无法轻易处理 Session	• 不规范的 HTTP 请求可能丢失 • 仅支持 HTTP 协议 • 无法轻易处理 Session • 个人项目，社区响应速度一般	• 底层较复杂，有一定的学习成本 • 会挤占线上应用的资源 • 定位偏回归测试，无回放倍数功能 • 有一定的开发工作量，维护成本较高 • 对工具本省的性能要求高 • 社区活跃度较低	• 有一定的学习成本 • 会挤占线上应用的资源 • 需要一定的开发工作量，维护成本较高 • 对工具本省的性能要求高 • 还不很成熟
成功案例	字节跳动（第二代放弃）	腾讯、京东	阿里巴巴	滴滴

极客时间

除了以上这些因素外，通常我们在选型工具的时候大概会考虑这么几个方面：

压力控制：指压测时流量倍数、TPS 的控制等；

数据驱动：大量的测试数据的参数化手段；

分布式支持：是否支持压力机集群；

测试报告：压测结果是否能够图形化展示，提供美观且丰富的测试报告；

性能开销：执行机开销、软件可靠性、执行效率、业务处理能力等。

选定工具，我们最好做下功能、性能、资源开销等方面的验证工作，做下选型 POC（验证）测试，确认选型是否符合我们的需求。

这里我们主要验证以下几个方面：

流量录制功能，录制的请求内容是否完整？数量是否正确？大流量下录制的资源开销情况如何？

流量过滤与改写功能，是否能够过滤指定 URL，是否能够改写指定参数？

流量回放功能，能否处理 Session 问题？

流量回放功能，是否能正常地回放流量文件？倍数功能是否正常有效？高倍数回放下资源开销如何？

综合对比评估后，最后我们选择了老牌流量回放工具 **GoReplay**，主要原因是简单、轻量、热度够、完全能满足我们目前项目的要求。

流量平台的建设

这里我再总结下全链路流量平台必须具备的能力：

能录制线上真实流量；

能实现海量数据的并发请求，并覆盖地域性的 CDN 边缘节点；

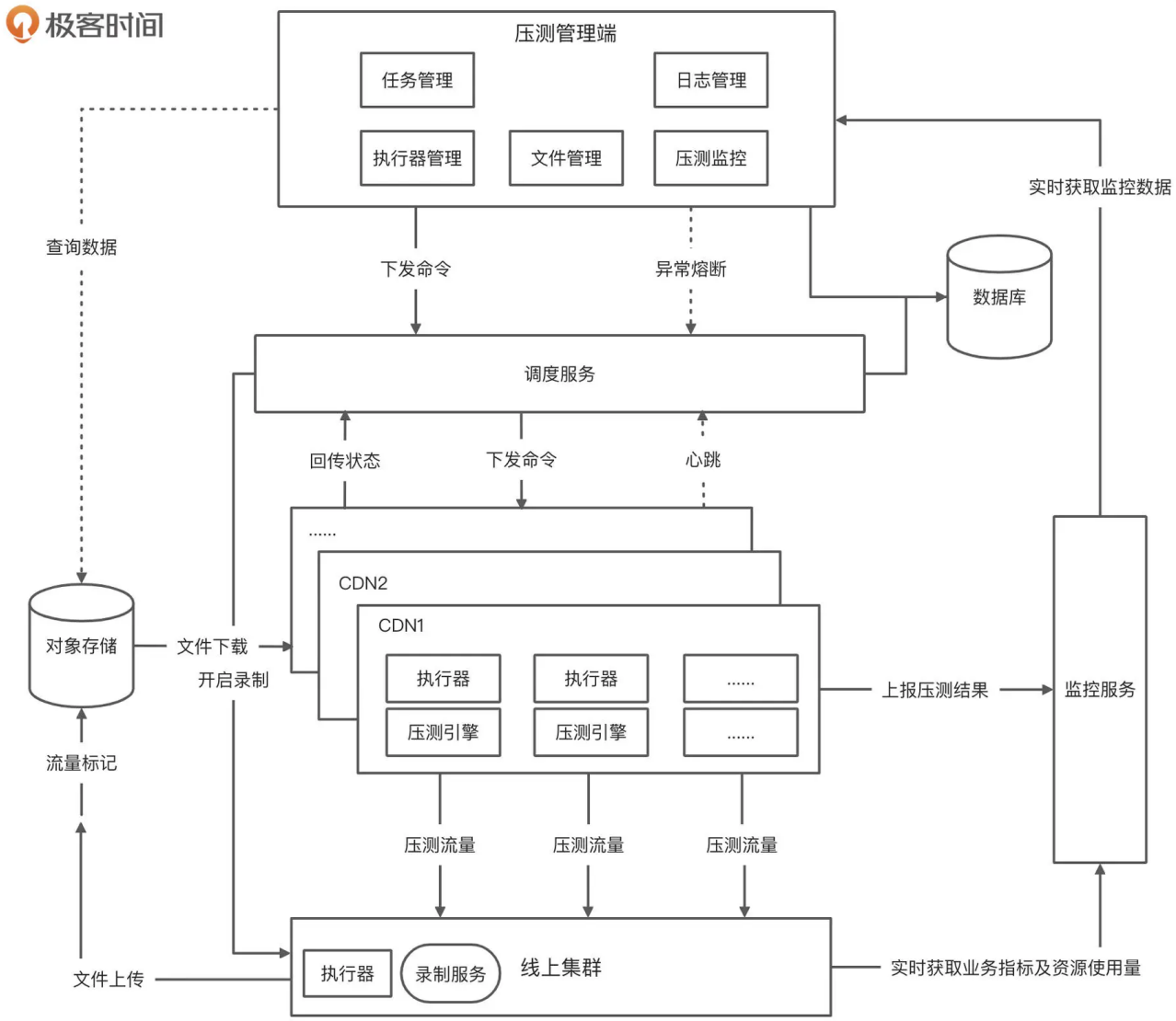
能支持常见协议的请求；

对线上尽量应用透明，也就是说无侵入性；

避免写请求的脏数据，压测流量能够被识别，方便压测后清理；

工具使用简单，能够满足压测实时监控，服务安全保护（过载熔断）。

那么，按照以上的能力需求，我画了一个比较典型的全链路流量平台架构设计图。



平台整体架构分为六大模块：

压测 web 管理端：负责流量文件的管理、压测任务的管理、压测过程中场景动态控制、日志管理、压测数据实时展示等；

调度服务：负责压力引擎调度、压测命令分发和执行器的管理等工作；

压测引擎：负责产生各种压测流量；

监控服务：实时统计压测结果数据，并监控各种资源及业务指标；

对象存储：存储流量文件，并支持快速查询、上传下载等功能；

录制服务：负责线上真实流量的复制，并支持对流量标记等功能。

这样的一个全链路流量平台基本上就可以覆盖大部分企业的需求了。在后续的课程中，我会就这里面的技术细节一一进行拆解。

总结

流量平台是一个全链路压测项目的引擎，如果你只是考虑某一个项目的快速压测，可能并不需要设计这么复杂并且重量级的流量平台。但对于一个大企业来说，全链路流量平台就显得极为重要了。因为它融合了整个公司线上压测的过程。

在全链路流量平台中，我强调了几个重点：保证流量的真实性、保证环境的真实性，它们都会对整全链路压测项目的质量起到关键作用。

我还带你梳理了目前市面主流的流量工具，当我们在做选型的时候，一定要牢记下面几项关键原则：

满足业务需求是第一准则；

工具成熟少踩坑；

要有高效解决问题的手段。

紧接着，我把一个全链路流量平台该有的能力以及需要做到什么地步，都给你梳理了一遍。希望能给你一些借鉴。

当然了，我更希望看到，你可以在后续的项目中尝试去设计一个完整的全链路流量平台。

思考题

学完这节课，请你思考两个问题：

1. 常见的构造流量有那几种方式？
2. 为什么我们一直要强调保证流量的真实性？

欢迎你在留言区与我交流讨论。当然了，你也可以把这节课分享给你身边的朋友，他们的一些想法或许会让你有更大的收获。我们下节课见！

分享给需要的人，Ta订阅后你可得 20 元现金奖励

生成海报并分享

赞 2 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 05 | 铺底数据：真实的压测数据应该做成什么样子？

下一篇 07 | 全栈监控：如何设计全栈监控策略？

精选留言

写留言

> 由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。